AD-A230 883

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. Agency Use Only *(Leave blank).*	2. Report Date.	3. Report Type and Dates Covered.
	27-31 March 1989	Symposium

4. Title and Subtitle.
Portable Intelligent Diagnostic Aids

5. Funding Numbers.
Program Element No. 25620N
Project No. 00101
Task No. 101
Accession No. DN257009

6. Author(s).
Randy Holland

7. Performing Organization Name(s) and Address(es).
Naval Ocean Research and Development Activity
Code 252
SSC, MS 39529-5004

8. Performing Organization Report Number.
PR 90:020:252

9. Sponsoring/Monitoring Agency Name(s) and Address(es).
Naval Sea Systems Command
CDR. Karl Kauffman PMS-411
Washington, DC 20362-5101

10. Sponsoring/Monitoring Agency Report Number.

11. Supplementary Notes.

12a. Distribution/Availability Statement.
Approved for public release;
Distribution is unlimited.

12b. Distribution Code.

13. Abstract *(Maximum 200 words).*

14. Subject Terms.
expert systems, ASW scenarios, artificial intelligence, maintenance advisors

15. Number of Pages.
7

16. Price Code.

17. Security Classification of Report.	18. Security Classification of This Page.	19. Security Classification of Abstract.	20. Limitation of Abstract.
Unclassified	Unclassified	Unclassified	

JCS SYMPOSIUM ON AI APPLICATIONS FOR MILITARY LOGISTICS

PORTABLE INTELLIGENT DIAGNOSTIC AIDS

By C. Randy Holland

NAVAL OCEAN RESEARCH AND DEVELOPMENT ACTIVITY

ABSTRACT

Effective troubleshooting of complex equipment typically requires large amounts of equipment, documentation, and training. The considerable burden this imposes on military resources must be reduced. Computer technology offers the possibility of untiring intelligent machines which compute and advise. As Diagnostic Tutors and Expert Aids, such machines should be effective force multipliers with minimal logistics impact. What constitutes a good diagnostic aid; the state of present achievability; the avoidance of development failure; and some future potentials are explored in this paper. Key characteristics of successful Expert Systems developments are beginning to come into focus. While the ideal diagnostic aid may rarely be achieved; practical, supportable aids are beginning to emerge from various laboratories. While our intelligent machines are still infants, even kids can do amazing things.

The Difficulty of Troubleshooting

Troubleshooting, that set of techniques for identifying faults in equipment, appears to be part art and part science. While logical troubleshooting procedures can be completely specified and reduced to a set of computer instructions; this set of instructions, blindly followed, will rarely outperform a highly experienced technician. But, it takes a great deal of training and experience to produce a highly experienced technician.

In addition, complex equipment, especially electronic equipment, requires the use of sophisticated test equipment or built-in test capability. Typically, an aspiring maintenance technician must first learn the fundamentals of mathematics and physics. Next he/she must learn how these fundamentals are applied in making tests. Then they learn structured use of test equipment for general troubleshooting. The next step is to learn as much as possible about each operational system being supported so that all the previous learning can be effectively used to keep things fixed.

But, all this training is still not enough to get the maintenance job done. The complexity of modern equipment, much of it electronic, far exceeds an individual's ability to remember how it works internally. Therefore, massive amounts of documentation are required that explain where and how and when to test, adjust, and/or replace items. So, in times past we have found the maintenance technician armed with manuals and test equipment; pawing through the innards of a system to find what is wrong and correct it.

More recently considerable progress has been made in

incorporating test equipment into the operational system itself. Typically, these built-in test functions return fault codes that can be interpreted and used in fault diagnosis. One 'new' problem in large systems is trying to remember all the fault codes or, at least, where they can be found in the documentation.

Computers - The Promised Solution

Once programed, computers don't forget. They are also very good at rapid testing and following long sequences of steps. They don't get tired or bored and, when properly structured, can virtually eliminate the need for cumbersome maintenance manuals. Computers have been used very successfully in automatic testing, system self-testing, and automatic fault detecting. As independent test devices and measurement support tools, computers have more than demonstrated their value. As intelligent diagnostic aids, they hold much promise; but also much concern.

Reliance on computers will continue to grow. What is the penalty if (or when) the computer fails? Do we revert back to general purpose test equipment (GPTE) and large stacks of maintenance manuals? As computers are used more and more, it would seem that proficient use of GPTE becomes even less likely. It is probable that near total reliance on computers mandates exceedingly high levels of reliability and availability. The computer technology trends of the last few years indicates that such very high levels can be obtained at affordable costs.

Characteristics of a Good Diagnostic Aid

If computer assistance is the future, then what characteristics must the diagnostic system possess? Other than high reliability, the system must be portable, smart, user friendly, fast, and perhaps teachable.

Portability means that the system can be handcarried by one individual aboard any military platform where it is needed. This does not mean that one system will meet all needs; but it does mean that each system must be convenient to used and perform in its assigned environment. Even more important than size is the need to survive the environment. Warfare occurs in nasty situations which fully tries men's souls and aggressively gnaws at equipment.

Smart computer systems do not guarantee friendly, usable systems. Each developed system must have the appropriate knowledge and the ability to communicate with the user. Communication is not just presenting information; it must also include the creation of understanding within the user. This goal requires careful attention to man-machine interface design; especially graphical presentations. Rather frequently we find scientists and engineers designing computer systems as though they will be used by other scientists and engineers. In the case of military operators, the system must not only be capable of solving time consuming tasks very well, but it must be very user friendly. A smart computer system must therefore solve, in an acceptable manner, all of the pertinent problems presented to it. It is

also desirable that unanticipated inputs produce outputs which are nonhazardous, if the system does not immediately recognized the inputs as being outside the scope of system capability.

Smart systems must also be computationally fast Processing speed requirements are directly related to the size of the knowledge base or search field. The larger the number of rules to be considered, the faster the computer must work. If a user must wait for an answer longer than he would expect to wait on a human expert, he will likely view the system as pretty dumb; or at least aggravating.

In reality, smart systems are never complete. There is almost always more that could be taught to them. The system must be initially designed with knowledge expansion in mind. Utilization will uncover new problems that ought to be included. Expandability will insure that the system is not soon relegated to the level of being helpful at times but certainly no expert.

What Can We Achieve Today?

The personal computer (PC) that you can set on your lap has really arrived. Within a few short years low power memories have grown from kilobits to megabits, and soon, to gigabits. Larger memories mean larger knowledge bases. Higher clock speeds mean faster processing. The trend is clearly and dramatically upward in performance with no end in sight. What computer technology cannot do today, it will do next month or next year, but soon. So take what I say about today's capabilities as a momentary snap shot which is already obsolete. On today's Navy standard issue 16 bit, 4 to 8 MHz machines we can handle a few hundred expert system rules; and do it in near real-time from a human's standpoint. A few hundred (but less than 1000) rules is sufficient to meet many higher level needs such as fault code interpretation but is inadequate for detailed troubleshooting at the component level for any but small systems. While the large memory of the PC can hold thousands of rules, the response time becomes totally unacceptable.

Processing speed is the great holdup today and is most limited by continuing dependence on von Neumann sequential processing architectures. Faster computers will help, but other measures should also play a part. The situation is not unlike that of an overworked individual. If you're going as fast as you can and you are given more data to process, the output will be delayed. We usually solve the problem by creating a team and dividing up the work. One can do the same with computers by adding more processors (CPUs) and distributing the load. While not as easily accomplished with computers, the work in distributed processing is promising. One can also divide up the data set into smaller 'chunks' in those case where all of the data does not have to be considered each time. This is usually true of expert systems. Review of only a small subset of the total data set definitely increases response time; but, great care must be exercised to insure that all appropriate data is in the reviewed subset.

PCs can, and some are, being made very rugged and reliable today. We have flown them in Navy aircraft, used then on many types of surface ships, and had them work well in submarines. I don't know how well they would work in the jungle but they do well in the Arctic once they get warmed up. I see no great technical difficulty or unusual expense in achieving practicable levels of ruggedness and reliability.

Ultimate usability of a smart system is decided by its friendliness. It can be very smart, fast, and always ready to work; but if it demands specialized knowledge from its user, or 'talks' down to him it will not likely be used much. By specialized knowledge, I mean insight into the internal workings of the system and how to string things together to make it perform. If the user must remember artificial commands, or must know which menu to call when, or what device contains the required data, the system usability will be limited to the few who remember such things well and are interested enough to learn them in the first place.

Talking down to the user is another potential problem. A system which assumes you know nothing about it and always tells you everything, tests one's patience. A system which uses unfamiliar jargon or wealthy words may confuse and/or insult. Good computer interfaces adapt to a wide range of users, permitting the novice to be guided step-by-step through the entire process and the experienced person to jump over routine items which are soon mastered from frequent use. Such adaptive systems must provide quick access to help and a return at any time to basics.

Project Expert

We have recently been building a demonstration maintenance advisor for a portion of a Navy surface ship sonar. Project Expert has as its goal the demonstration of improved fault diagnosis in an area of complex analog circuitry where extensive built-in fault detection is typically not feasible. The approach is based on using what fault codes are provided at a high level followed by conventional troubleshooting with general purpose test equipment. EXPERT, the diagnostic aid, will reside in a Hewlett Packard 9807 portable computer having the following specifications:

Motorola 68000 microprocessor operating at 8 MHz
16 bit Graphics processor (32K of RAM)
HP-UX (UNIX System V) Operating System
1.0 Mbyte of internal RAM expandable to 2.6 Mbytes
Single 710 Kbyte 3.5 inch floppy diskette
9-inch electroluminescent display (256 x 512 pixels)
Full function keyboard with numeric pad
Built-in Thinkjet dot matrix printer
Weighs 25 pounds and measures 7 x 13 x 16 inches

Originally, we intended to develop a maintenance advisor which would imitate an electronic technician who was well

versed in troubleshooting techniques and quite knowledgeable on the sonar. We found the Navy Center for Applied Research in Artificial Intelligence had developed a shell called FIS (Fault Identification System) and that it appeared to be just what we needed. Written in LISP, it supported testing down to any level using probability of likely failure functions that included previous experience. This shell met most all of our needs, so we began coding in the details regarding connected components and subsystems and the signal flow paths.

We were a bit surprised when the number of rules rather quickly exceeded 1000; but we were really shocked when the number exceeded 3000 for just the small part of the sonar system we selected for demonstration. When we tried to run EXPERT interactively, it ran horribly slow; too slow to be of real value. It was clear that we had to find another way to organize the knowledge base so that the system did not need to exhaustively search all the rules each time the next piece of data was input. We fairly quickly decided that 'divide and conquer' offered the best possibility for recovery; much like prioritizing work into stacks and working on the most important stack first.

Eventually, we settled upon the concept of a high level expert supported by a number of lower level expert modules. The high level expert uses the fault codes of the limited built-in test capability to determine the best area for beginning troubleshooting. The appropriate lower level expert module or modules are called and the rules pertaining only to the limited knowledge of each are examined. Each lower level expert module accepts information from the single high level expert and returns results to it. Since the high level expert acts as the coordinator of all activity, it provides all test and information continuity management.

We have yet to determine under what expanded conditions the system will once again become unacceptably slow. I suspect that the system will become too slow when the size of one or more lower level modules grows beyond some critical value. When that occurs, additional hierarchical layering will be further explored. I also suspect that parallel processing will help. If there are three levels of expertise and the top two are computers, then the highest level expert could activate more than one trouble shooter simultaneously. This arrangement would be akin to having more than one maintenance technician working on the system at the same time.

Avoiding Failures

Expert maintenance advisors are seemingly simple in concept; but from our experience not so simple to achieve in practice. The principal problem is usually trying to accomplish too much with a single expert/single CPU system. One thing you learn quickly in developing expert maintenance systems is the tremendous amount of data that humans routinely process. We have found the following guidelines to be helpful:

a. Carefully define what performance is really needed.

b. Avoid 'neat' bells and whistles.
c. Watch the size of the knowledge base/rule set.
d. Build early versions and test often.

Thoughtful definition of required performance and how it will be verified is the most critical step in building any expert system; and, the step most often glossed over. If this step is well executed, the probability of a successful expert system development will increase dramatically. A useful scheme we have developed consists of defining two performance goals. The first goal defines the minimum acceptable performance. If the system doesn't meet or exceed each and every one of these requirements it isn't acceptable. The second goal defines what we would really like the system to accomplish, without any frills. Goal two is a solid full performance, as opposed to a minimum performance, definition. These two goals become the measure by which project progress and ultimate success are measured. Goal one must be attained. If goal two is not attained, a usable product still emerges. If goal two is achieved, a very good product emerges. If goal two is achieved and time and money remain, a superb product is possible.

With time and money remaining, some 'bells and whistles' can be added to further increase the value of the expert system or additional testing can be performed to more fully explore the total capabilities of the system. Typically, additional time and money will not be available but a useful product will emerge because goal one was accomplished. By avoiding bells and whistles early on you will not squander precious development resources on nice but not necessary system features.

One key to danger ahead is a continually expanding knowledge or rule base. Speed of response is directly related to the size of the rule set that must be considered. Rule sets larger than a few hundred rules start running slow on today's PCs. Rule sets of a few thousand rules run unacceptably slow. An experienced knowledge engineer should try to estimate the probable size of the rule set early in the project planning phase. If you find that the rule set grows larger than expected you are in trouble already and need to either reduce the scope of performance or divide things up into a hierarchy

The *proof of success* is in the demonstration of achieved performance. Until you run the expert system, it is next to impossible to know how it will perform. Because of the ease with which they can be upgraded and expanded, partial expert systems should be constructed early and refined. The process is not unlike teaching a new subject to a human. You would not send an individual to school for a whole year before giving any tests. So why work a year on an expert system before testing its level of knowledge? Frequent testing is a good way to determine knowledge increase. Frequent testing, improvement and retesting of an expert system is the surest way to know where you are in goal attainment.

Future Possibilities

There is no doubt that PCs will continue to get faster and more capable. Expert systems that run real slow on today's computers will operate in near real-time on tomorrow's machines. I expect to see a very large number of computer based expert aids and advisors developed over the next ten years; some pocket sized and some lap sized, but almost all portable and very user friendly. There will be a large impact in the consumer market which once again will benefit the military through low cost equipment to demonstrate new concepts. Many devices could be multiprocessor based with some processors acting in parallel and some operating semi-independently.

With all the wondrous improvement that will surely come in the computer technology area, the greatest need will be in the human-computer (man-machine) interface. We must not require users of intelligent systems to become more knowledgeable to keep up with system growth. Having always to learn more to use the next smarter system will slow acceptance and usage benefits in kind. Intelligent systems must make it easier to use then as they grow in capability if we are to provide expertise extensively throughout the armed forces. The potential benefits are powerful inducements to succeed.

And, what about systems that learn on their own. I doubt we will see anything impressive very soon. The human is so very far ahead of the computer in many areas and we know so little about how we function mentally. But, who knows, miracles do happen!